Knowledge-Based Systems 275 (2023) 110725

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Robust anomaly detection for multivariate time series through temporal GCNs and attention-based VAE

Yunfei Shi [a], Bin Wang [a], Yanwei Yu [a,*], Xianfeng Tang [b], Chao Huang [c], Junyu Dong [a]

[a] College of Computer Science and Technology, Ocean University of China, Qingdao, Shandong 266100, China
[b] Amazon Search, Palo Alto, CA, USA
[c] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong, China

ARTICLE INFO

Article history:
Received 12 June 2022
Received in revised form 15 March 2023
Accepted 10 June 2023
Available online 14 June 2023

Keywords:
Anomaly detection
Multivariate time series
Unsupervised learning
Graph neural networks
Variational auto-encoder

ABSTRACT

Anomaly detection on multivariate time series (MTS) is of great importance in both data mining research and industrial applications. While a handful of anomaly detection models are developed for MTS data, most of them either ignore the potential correlations between different variables or overlook the different importance of variables at each time period in MTS, which leads to poor accuracy in anomaly detection. In this paper, we propose a novel unsupervised **MU**ltivariate **T**ime series **AN**omaly de**T**ection framework (**MUTANT**), which simultaneously models the correlations between variables and the importance of variables at each time period. Specifically, we construct a feature graph for variables in each time window and perform graph convolutional network (GCN) to learn embeddings for all variables, which effectively captures the time-varying correlations between variables in MTS. Then, we propose an attention-based reconstruction model to learn robust latent representations to capture normal patterns of MTS by modeling the importance of variables based on time dependencies along with time dimension. Our evaluation experiments are conducted on four real-life datasets from different industrial domains. Experimental results show that MUTANT significantly outperforms state-of-the-art MTS anomaly detection methods, achieving an average anomaly detection *F1-score* higher than 0.96. The source code is available at https://github.com/Coac-syf/MUTANT.

1. Introduction

Anomaly detection has been widely studied in different domains [1] (*e.g.*, log messages, time series, graphs, etc.), aiming at finding which instances significantly deviate from the other observations in the same dataset [2]. In this work, we mainly study the problem of anomaly detection in Multivariate Time Series (MTS for short hereafter), which has attracted much attention in data mining community. A large amount of MTS data is generated by sensors in industrial devices, such as server machines [3], spacecrafts [4], robot-assisted systems [5]. Anomaly detection for MTS is widely used to monitor the status of the devices in the application domain of manufacturing industry and IT systems [6–9].

Generally, it is preferred to identify entity anomalies at the entity-level directly using MTS, rather than at the metric-level using univariate time series [4,5,7,8,10]. In this work, we focus on detecting the overall anomalies of the MTS of each monitored entity. However, detecting anomalies at entity-level is very challenging. Firstly, due to the lack of labeled anomalies in historical data, and the unpredictable and highly varied nature of anomalies, supervised learning methods are infeasible. Secondly, for a complex real-world system, several monitoring metrics are often related to each other due to their intrinsic connections (*e.g.*, related sensors in a water treatment plant), thus an incident at an entity type may cause anomalies in multiple metrics. Single univariate time series cannot reveal information on these global properties. Therefore, anomaly detection on multiple univariate time series does not perform well for MTS. Thirdly, there may exist strong temporal dependencies in MTS data. Due to this reason, many classical unsupervised approaches, *e.g.*, distance/density-based methods [11–14], and density estimation methods [15–18], perform poorly since they cannot capture temporal dependencies along with time dimension.

Previous studies towards this end, have made significant efforts on anomaly detection for MTS. For instance, Hundman et al. [4] leverage LSTM to detect anomalies in multivariate time-series metrics of spacecraft based on prediction error. Su et al. [8] propose a stochastic recurrent neural network, OmniAnomaly, which captures the normal patterns of MTS by learning data

* Corresponding author.
E-mail addresses: shiyunfei@stu.ouc.edu.cn (Y. Shi), wangbin9545@ouc.edu.cn (B. Wang), yuyanwei@ouc.edu.cn (Y. Yu), xianft@amazon.com (X. Tang), chaohuang75@gmail.com (C. Huang), dongjunyu@ouc.edu.cn (J. Dong).

https://doi.org/10.1016/j.knosys.2023.110725
0950-7051/

representations through stochastic latent variables. DAEMON [10] combines the autoencoder and adversarial training and designs two groups of generators and discriminators to learn the normal pattern of MTS and thereafter uses the reconstruction error to detect anomalies. InterFusion [19] simultaneously models the intermetric and temporal dependency for MTS, which learns the normal patterns in MTS through hierarchical VAE with two stochastic latent variables. GDN [20] uses an attention-based graph neural network (GNN) to learn the relationship between different sensors, and additionally provide explainability for the detected anomalies using attention weights. However, these works either use RNN models to model time series while *ignore the potential inter-relationships between variables (metrics) in MTS* [4,5,7,8], or adopt GNNs to capture multivariate correlations explicitly but *ignore the different importance of variables at each time period in MTS* [20,21]. In this work, we aim to learn robust latent representations to capture normal patterns in MTS, considering both the correlations between variables and the importance of variables based on temporal dependencies at each time period.

Nevertheless, there are still two major challenges that remain to be solved for our goal. *The first challenge is how to learn the time-varying correlations between variables in MTS.* MTS is comprised of a group of univariate time series (*i.e.*, variables), and some variables in MTS often show an inconsistent pattern with other variables due to their intrinsic connections in complex real-world systems. Consequently, in the MTS anomaly detection systems, it is very necessary to consider the correlations between variables. *The second challenge is how to capture the time dependencies on time series and learn the importance of variables for reconstruction-based anomaly detection.* Although there exist correlations between variables, the importance of different variables to anomaly detection is different, and the importance is also different in different time periods. That is, the importance of each variable to reconstruction-based anomaly detection is also time-dependent in MTS. None of existing methods consider modeling the importance of variables along time dimensions in MTS anomaly detection.

To tackle the aforementioned challenges, we propose a novel unsupervised **MU**ltivariate **T**ime series **AN**omaly de**T**ection framework (**MUTANT**) based on GCN and Variational Auto-Encoder (VAE) architecture. Specifically, we first construct a feature graph based on variables' features for each time window in MTS, and then we perform Graph Convolutional Network (GCN) to learn the embedding vectors for all variables in each time window, which effectively captures the time-varying correlations between variables in MTS. In addition, we design an attention-based reconstruction model, consisting of an LSTM-based attention module that learns the importance of variables in each time window based on time dependencies in the time dimension and a VAE module that learns the latent intrinsic representation for each observation to capture "normal patterns" of MTS. Furthermore, we use end-to-end training to optimize our model by a joint learning objective function. Our experimental evaluation on four benchmark datasets demonstrates that our proposed MUTANT significantly performs better than state-of-the-art MTS anomaly detection methods, achieving up to 7.18% improvement in terms of *F1-score*.

We highlight the key contributions of this work as follows:

- We propose a novel reconstruction-based MTS anomaly detection framework, considering both the time-varying correlations between variables and the importance of variables based on temporal dependencies at each time period in MTS.
- We propose an attention-based reconstruction model that jointly learns the importance of variables via the proposed LSTM-based attention module and the robust representation for observations via VAE to capture "normal patterns" of MTS.
- We conduct extensive experiments on four real-life MTS datasets to demonstrate the superiority of our model when competing with state-of-the-art baselines, achieving up to 7.18% improvement in terms of the F1-score. Furthermore, the ablation study also verifies the rationality of our designed sub-modules, and robustness evaluation with respect to noise is also investigated.

2. Related work

Anomaly detection on univariate time series. Anomaly detection in time series is a challenging and interesting task that has been studied extensively [22]. Yahoo EGADS [3] is a general and scalable framework for detecting anomalies in large-scale time series by using a combination of anomaly detection and forecasted modules with an anomaly filtering layer. Twitter [23] proposes a method called the Seasonal Hybrid Extreme Study Deviation test (S-H-ESD), which can detect both local and global anomalies in time series. In 2017, Google [24] tested the performance of deep learning models (including DNNs, RNNs, and LSTMs) for anomaly detection on their datasets and achieved the expected results. The rapid development of neural networks provides a solid foundation for improving the accuracy of anomaly detection. DSPOT [25] uses extreme value theory to detect anomalies in streaming univariate time series without making assumptions about the distribution of the raw data or manually setting thresholds. LAKE [16] and ADAF [26] are proposed to detect anomalies in high-dimensional data using layer-constrained VAE and autoregressive flow models, respectively. Donut [27] is a VAE-based unsupervised model that detects anomalies in seasonal KPIs. SR-CNN [28] applies Spectral Residual (SR) in the domain of visual saliency to anomaly detection and combines with Convolutional Neural Networks (CNN) to improve the model's performance. Nevertheless, these methods focus on unit time series and cannot be directly applied to anomaly detection on multivariate time series.

Anomaly detection on multivariate time series. To detect spacecraft anomalies, LSTM-NDT [4] applies LSTM for MTS prediction, and then determines anomalies using prediction error. EncDnc-AE [7] is an LSTM-based encoder–decoder model to obtain latent patterns of multi-sensor time series by reconstructing normal data, then identifying anomalies based on reconstruction errors. DAGMM [15] combines deep AE and Gaussian Mixture Model (GMM) to detect anomalies. But this method does not involve the time dependency of the data, so it is only suitable for multivariate variables (not MTS). USAD [29] is a model with one encoder and two decoders, which uses the idea of adversarial training to train the model, so as to increase the gap between normal data and abnormal data. DAEMON [10] combines the autoencoder and adversarial training, and designs two groups of generators and discriminators to obtain the normal patterns of MTS, and thereafter uses the reconstruction error to detect anomalies. These two methods expect to widen the gap between normal data and abnormal with the idea of adversarial training but do not take into account the time dependence of the sequence.

In order to solve this problem, LSTM-VAE [5] replaces the feed-forward network in VAE with LSTM to obtain temporal dependencies. Similarly, OmniAnomaly [8] combines GRU with VAE and uses stochastic variable connection and planar normalizing flow to improve the performance of the model, finally determining anomalies according to the reconstruction probabilities. MSCRED [9] constructs a multi-scale representation matrix to capture system states at multiple levels and uses attention-based ConvLSTM to capture temporal relationships. And MAD-GAN [6] uses LSTM-RNN as the base model to capture temporal dependencies and embeds them into the framework of GAN while utilizing

the generator and discriminator of GAN to detect anomalies. However, these methods only focus on the most basic characteristics of time series data, that is, the time dependence of data, but ignore the relationship between variables and the importance of different variables.

InterFusion [19] uses hierarchical VAE with two stochastic latent variables to obtain the normal patterns of MTS. AMSL [30] is a self-supervised MTS anomaly detection model, which improves the generalization ability of the model through a convolutional autoencoder. TimeAutoAD [31] is also a self-supervised detection model. This method proposes three strategies to generate pseudo-negative time series based on training data, and distinguish the generated data from the original data by comparing the loss, so as to improve the monitoring performance of the model. ELM-AD [32] is a cluster-based anomaly detection method for multivariate time series. This method combines the multivariate ELM-MI framework with the dynamic kernel selection method and determines the kernel in ELM-MI through clustering. Nevertheless, the above methods treat all variables in MTS equally and fail to capture the time-varying correlations between variables in detecting anomalies.

Graph Neural Networks. GNNs have achieved great success as a common model for processing graph-structured data. In general, the theory of GNNs believes that the feature of the current node is affected by the feature of its neighbor nodes. GCN [33] obtains the representation of a node by aggregating representations of its one-step neighbors. Based on this idea, graph attention network (GATs) [34] assigns different weights to different neighbors through an attention function when aggregating the representation of neighbors, so as to reflect the different influences of different neighbors on the current node. Related variants have also been successfully applied to anomaly detection of MTS, for example, [21] utilizes two parallel graph attention (GAT) layers to capture temporal and spatial dependencies respectively. GDN [20] is also a method based on graph attention neural network, which mainly obtains the relationship between different sensors, and judges abnormality according to whether it deviates from these relationships. GNN-DTAN [35] is a method based on a graph neural network, it uses the graph construction module in the model to extract the relationship between features. The trained model is then used to predict the data, and the anomaly score between the predicted and actual values is calculated. Although graph-based methods consider the relationship between variables, they ignore that the effects of different variables are different for anomaly detection.

3. Problem statement

Definition 1 (*Multivariate Time Series, or MTS*)**.** Multivariate time series is a time series of successive observations which are collected at equal-space timestamps, defined as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where n is the length of $\mathbf{X}$, each observation $\mathbf{x}_t \in \mathbb{R}^m$ is a m-dimensional vector at time point $t(t \leq n)$: $\mathbf{x}_t = \{x_t^1, x_t^2, \ldots, x_t^m\}$, and m is the number of variables [8].

Given an MTS $\mathbf{X}$ as training input, the objective of unsupervised MTS anomaly detection is to identify whether an unseen observation $\mathbf{x}_t$ $(t > n)$ is anomalous or not. For time series modeling, historical values are beneficial for understanding the current time point. Therefore, we define a time window of length τ at given time point t: $W_t = \{\mathbf{x}_{t-\tau+1}, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t\}$. The original time series $\mathbf{X}$ can be transformed into a sequence of time windows $\mathcal{W} = \{W_1, W_2, \ldots, W_n\}$ to be used as training input, and a time window W_t instead of observation $\mathbf{x}_t$ is used to calculate the anomaly score.

Based on the above definitions, we next formally define our studied problem as follow:

Table 1
Key notations and their definitions.

Notation	Definition
$\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$	A multivariate time series
$\mathbf{x}_t \in \mathbb{R}^m$	The observation at time t
W_t	The time window at time t
$\mathbf{X}^i$	All values of ith variable in $\mathbf{X}$
n	The length of $\mathbf{X}$
m	The number of variables
τ	The length of time window
$\mathcal{G}_t$	The feature graph of W_t
$\mathbf{A}_t$	The adjacency matrix of $\mathcal{G}_t$
$\mathbf{H}_t^{(l)}$, $\tilde{\mathbf{x}}_t$	The learned embeddings of W_t
$\tilde{\mathbf{x}}_t$	The weighted embeddings of W_t
$\hat{\mathbf{x}}_t$	The reconstructed vector

Problem 1 (*MTS Anomaly Detection*)**.** Given an MTS $\mathbf{X}$, the goal of our anomaly detection problem is to calculate an anomaly score for an unseen observation $\mathbf{x}_t$ $(t > n)$, and then a binary label y_t (*e.g.*, $y_t = 1$ indicating an anomaly, 0 for not) is determined by compared against a threshold.

The key notations used in this paper are summarized in Table 1.

4. Methodology

In this section, we present the details of our proposed MUTANT (as shown in Fig. 1), consisting of two key components: (i) *Temporal GCN*, and (ii) *Attention-based reconstruction module*. *Temporal GCN* aims to obtain the representation vectors for variables, which implies the potential connections between different variables in each time window. *Attention-based reconstruction module* attempts to capture the normal patterns of MTS by learning their robust latent representations with *LSTM-based attention* and *VAE*, and uses the reconstruction errors to determine anomalies.

Although GCN, LSTM, and VAE are common models in anomaly detection, there is no previous method to integrate the models together, so as to achieve the purpose of considering the time dependence of time series, the relationship between variables, and giving different weights to different variables. Most importantly, from the subsequent experimental results, the performance of MUTANT is significantly better than that of all state-of-the-art MTS anomaly detection methods on the evaluated real-world datasets.

4.1. Temporal graph convolutional network

MTS is usually continuously collected by multiple sensors, and at the same time or interval, variables collected from different sensors are often related. For example, in a medical monitoring system, the multiple sensors on the same patient are correlative. When the patient is in a sudden condition, they may change drastically at the same time. If the connections between them can be expressed and used in anomaly detection, it will naturally be helpful to detect anomalies.

In this work, we try to build the connections between variables in MTS and obtain more reasonable representations for all variables in each time window according to such connections for anomaly detection. GCNs have been widely used in graph representation learning and have achieved great success [36–39]. To better capture the connections between different variables, we introduce the GCN model to learn the representations for variables in each time window.

To adapt to the GCN model, we first use k-NN method to construct a feature graph $\mathcal{G}_t = \{\mathcal{V}, \mathcal{E}\}$ for each time window W_t

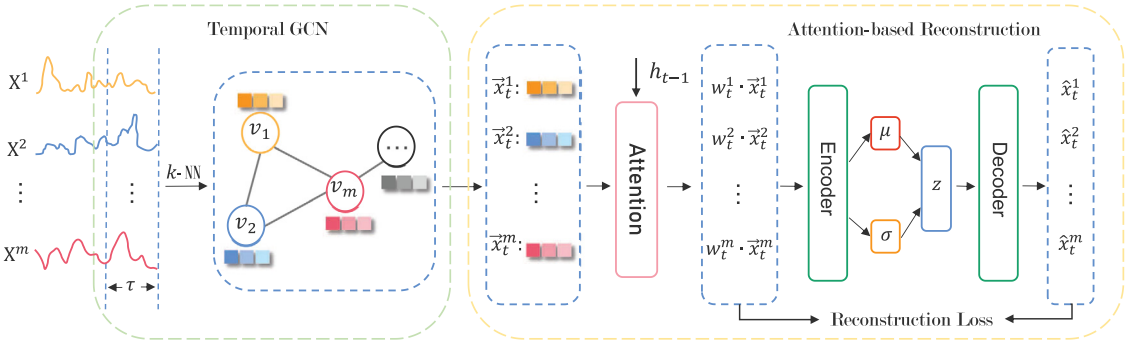


Fig. 1. The overview of the proposed MUTANT.

in $\mathbf{X}$, where each node $v_i \in \mathcal{V}$ is a variable, and each variable is connected with its most related k variables by an edge. Specifically, the value of each variable v_i in the current time window is regarded as its features, *i.e.*, $f_t^i = \{x_{t-\tau+1}^i, \ldots, x_{t-1}^i, x_t^i\}$. We first calculate the correlation coefficient matrix $\rho \in \mathbb{R}^{m\times m}$ among m variables in the feature space. In this paper, we employ the Pearson correlation coefficient (Eq. (1)), which is a popular way to obtain the correlation coefficient between two vectors.

$$\rho_{i,j} = \frac{Cov(f_t^i, f_t^j)}{\sqrt{Var[f_t^i]Var[f_t^j]}}, \tag{1}$$

where $Cov(f_t^i, f_t^j)$ is the covariance of f_t^i and f_t^j, $Var[f_t^i]$ is the variance of f_t^i, and $Var[f_t^j]$ is the variance of f_t^j.

We use adjacency matrix $\mathbf{A}_t \in \mathbb{R}^{m\times m}$ to represent the graph $\mathcal{G}_t$, where $\mathbf{A}_t^{i,j} = 1$ denotes that there is an edge between nodes v_i and v_j.

To capture the potential inter-relationships between variables, we employ graph convolution operation on each graph $\mathcal{G}_t$ to aggregate the features of variables from their neighborhood variables. Following [33], we perform the following layer-wise propagation rule in a multi-layer convolution network:

$$\mathbf{H}_t^{(l+1)} = ReLU(\tilde{\mathbf{D}}_t^{-\frac{1}{2}}\tilde{\mathbf{A}}_t\tilde{\mathbf{D}}_t^{-\frac{1}{2}}\mathbf{H}_t^{(l)}\mathbf{W}_t^{(l)}), \tag{2}$$

where $\mathbf{W}_t^{(l)}$ is a layer-specific trainable weight matrix, $\tilde{\mathbf{A}}_t = \mathbf{A}_t + \mathbf{I}$, and $\tilde{\mathbf{D}}_t^{ii} = \sum_j \tilde{\mathbf{A}}_t^{ij}$. $\mathbf{H}_t^{(0)} = W_t$, and $\mathbf{H}_t^{(l)} \in \mathbf{R}^{m\times d}$ is the output of lth layer for time window W_t, where d is the embedding dimension. The final output of temporal GCN is the time-varying embedding vectors of all variables in each time window W_t, denoted by $\vec{\mathbf{x}}_t = \mathbf{H}_t^{(l)}$.

4.2. Attention-based reconstruction module

After obtaining the embedding vector of each variable, we use a reconstruction model to better obtain the essential characteristics contained in the MTS for anomaly detection. This module mainly includes two parts: LSTM-based attention and VAE-based reconstruction module.

4.2.1. LSTM-based attention

Although an MTS includes multiple variables, the importance of each variable is definitely different for anomaly detection. And the importance of variables on different time windows may also be different. In addition, the time dependencies of variables on time series are essential for anomaly detection.

To capture the time dependence on MTS and learn the importance of different variables in different time windows, we propose an LSTM-based attention mechanism to achieve this goal. This is because many previous works have proved that LSTM can successfully obtain the time dependence of time series. Compared with other advanced attention models, the attention mechanism based on LSTM is simpler and more efficient. It can greatly reduce the training time and complete the function of weighting variables. As shown in Fig. 2, in each LSTM unit, we take the embedding vectors of variables in the time window as input and obtain the weights of variables in the current time window through a linear layer and a softmax layer. Then, we use the weighted embedding vectors as the input of the LSTM unit to obtain the weights of variables in the next time window. Each LSTM unit has two transmission states c_t and h_t, and c_t is limited by three gates, *i.e.*, forget gate f_t, input gate i_t and output gate o_t.

The formulation of LSTM-based attention is as follows:

$$f_t = \sigma(\mathbb{W}_f[h_{t-1}; \tilde{\mathbf{x}}_t] + b_f), \tag{3}$$

$$i_t = \sigma(\mathbb{W}_i[h_{t-1}; \tilde{\mathbf{x}}_t] + b_i), \tag{4}$$

$$o_t = \sigma(\mathbb{W}_o[h_{t-1}; \tilde{\mathbf{x}}_t] + b_o), \tag{5}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(\mathbb{W}_c[h_{t-1}; \tilde{\mathbf{x}}_t] + b_c), \tag{6}$$

$$h_t = o_t \odot \tanh(c_t), \tag{7}$$

where $[;]$ denotes concatenation operation, h_{t-1} is the previous hidden state, $\tilde{\mathbf{x}}_t$ is the current input, $\sigma(\cdot)$ is sigmoid function, $\odot$ is element-wise multiplication, and $\mathbb{W}_f$, $\mathbb{W}_i$, $\mathbb{W}_o$, $\mathbb{W}_c$ and b_f, b_i, b_o, b_c are learnable parameters.

More specifically, in each LSTM unit, we first concatenate h_{t-1}, c_{t-1} and $\vec{\tilde{\mathbf{x}}}_t$, and feed them into the linear layer to obtain $\breve{w}_t$. Then, after softmax layer conversion, the final weight w_t for all variables is obtained. The formulation of this process is:

$$\breve{w}_t^i = \tanh(\mathbb{W}[h_{t-1}; c_{t-1}; \vec{\tilde{x}}_t^i] + b), \tag{8}$$

$$w_t^i = \frac{exp(\breve{w}_t^i)}{\sum_{j=1}^m exp(\breve{w}_t^j)}, \tag{9}$$

$$\tilde{x}_t^i = \vec{x}_t^{\,i} \cdot w_t^i, \tag{10}$$

where $\vec{\tilde{\mathbf{x}}}_t^i$ is the embedding vector of ith variable in time windows W_t, and $\tilde{\mathbf{x}}_t = \{\tilde{x}_t^1, \tilde{x}_t^2, \ldots, \tilde{x}_t^m\}$ is the output of LSTM-based attention multiplying the embeddings of the variables by the corresponding weights. Afterwards, h_{t-1}, c_{t-1} and $\tilde{\mathbf{x}}_t$ are fed into LSTM unit to obtain the c_t, h_t for the next time window.

Since the input of LSTM units is the embedding vectors of the consecutive time windows, the LSTM-based attention module can capture the time dependencies on MTS and convert the dependencies into the weights of variables in different time windows, so that different variables play different roles in anomaly detection.

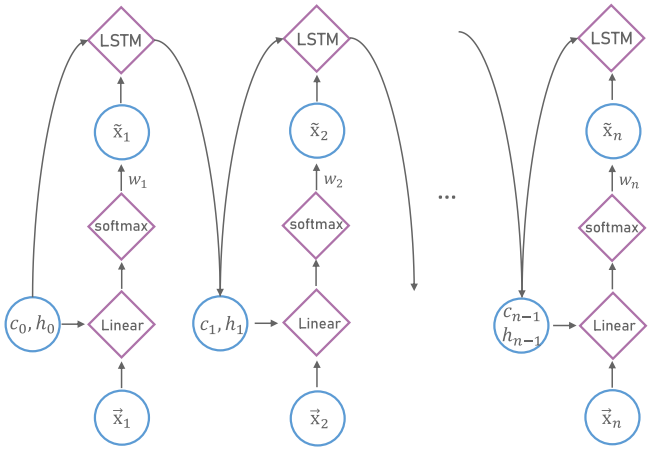


Fig. 2. The architecture of LSTM-based attention module.

4.2.2. VAE-based reconstruction module

Although we use temporal GCN to obtain the embeddings of variables in each window and employ LSTM-based attention to capture the time dependencies in the time dimension, for unsupervised anomaly detection, how to measure the difference between normal samples and potential abnormal samples is vital. That is, how determining whether a sample differs significantly from most samples in $\mathbf{X}$ is crucial for anomaly detection.

VAE [40] has been widely applied in MTS anomaly detection models [4,8,10] due to its ability to obtain latent patterns of high-dimensional data. In this work, we also leverage VAE to simultaneously train the reconstruction module and guide the learning of variable weights in LSTM-based attention.

VAE compresses high-dimensional input $\tilde{\mathbf{x}}_t$ into low-dimensional latent representation $\mathbf{z}_t$ by dimensionality reduction, and then reconstructs $\tilde{\mathbf{x}}_t$ by $\mathbf{z}_t$. Assuming that $\mathbf{z}_t$ obeys the prior distribution $p_\theta(\mathbf{z}_t)$, $\tilde{\mathbf{x}}_t$ can be sampled from the posterior distribution $p_\theta(\tilde{\mathbf{x}}_t|\mathbf{z}_t)$. However, it is very challenging to calculate $p_\theta(\tilde{\mathbf{x}}_t|\mathbf{z}_t)$ accurately, so VAE approximates it with an inference network $q_\phi(\mathbf{z}_t|\tilde{\mathbf{x}}_t)$, where θ and ϕ are the parameters in the generation network and inference network, respectively.

Like most VAE training methods, this work also use Stochastic Gradient Variational Bayes (SGVB) [41] to train the parameters in the VAE by maximizing the evidence of lower bound (ELBO), and the reconstruction-based loss function $\mathcal{L}_{re}(\tilde{\mathbf{x}}_t)$ for input $\tilde{\mathbf{x}}_t$ is:

$$\begin{aligned}\mathcal{L}_{re}(\tilde{\mathbf{x}}_t) = &-\mathbb{E}_{q_\phi(\mathbf{z}_t|\tilde{\mathbf{x}}_t)}[\log(p_\theta(\tilde{\mathbf{x}}_t|\mathbf{z}_t))] \\ &+ D_{KL}[q_\phi(\mathbf{z}_t|\tilde{\mathbf{x}}_t)||p_\theta(\mathbf{z}_t)].\end{aligned} \tag{11}$$

The first term is the reconstruction of $\tilde{\mathbf{x}}_t$ by maximizing the log-likelihood $\log p_\theta(\tilde{\mathbf{x}}_t|\mathbf{z}_t)$ with sampling from $q_\phi(\mathbf{z}_t|\tilde{\mathbf{x}}_t)$. The second term achieves regularization of latent variable $\mathbf{z}_t$ by minimizing the Kullback–Leibler (KL) divergence between the approximate posterior and the prior of the latent variable.

As shown in Fig. 1, we input $\tilde{\mathbf{x}}_t$ into VAE to reduce the dimension of variable embeddings. After the dimensionality reduction of the encoder, we get the latent representation $\mathbf{z}_t$, and through the decoder, we obtain the reconstructed value $\hat{\mathbf{x}}_t$. Then we use the reconstruction of $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{x}}_t$ to train VAE model.

4.3. Offline model training

To achieve the best detection performance, we implement end-to-end training for our model by designing a joint learning objective function. Below we formally define the joint learning objective function to obtain the result of global optimization.

First, for unsupervised learning, we use the following binary cross-entropy loss function through negative sampling to optimize the temporal GCN model:

$$\begin{aligned}\mathcal{L}_{GCN} = &-\sum_{(i,j)\in\Omega} \log\sigma(\langle(\vec{x}_t^{\,i})^{\mathsf{T}}, \vec{x}_t^{\,j}\rangle) \\ &-\sum_{(i',j')\in\Omega^-} \log\sigma(-\langle(\vec{x}_t^{\,i'})^{\mathsf{T}}, \vec{x}_t^{\,j'}\rangle),\end{aligned} \tag{12}$$

where $\vec{x}_t^{\,i}$ is the embedding vector of ith variable in time window W_t, T denotes matrix transposition, $\langle,\rangle$ can be any vector similarity measure function (*e.g.*, inner product), Ω is the set of positive node pairs in $\mathcal{G}_t$, and Ω^- is the set of sampled negative node pairs. That is, if $i, j \in \Omega^-$, then $\mathbf{A}_t^{i,j} = 0$. Our purpose is to maximize the similarities between the node embeddings in the positive samples and minimize the similarities between the node embeddings in the negative samples simultaneously.

Second, we use the Eq. (11) to train our reconstruction module. Finally, we combine $\mathcal{L}_{GCN}$ and $\mathcal{L}_{re}$ to jointly train our model through hyperparameter α, which is used to balance the importance of representation learning and reconstruction model. The joint objective function of our model is:

$$\mathcal{L}_{joint}(\Theta) = \mathcal{L}_{GCN} + \alpha\mathcal{L}_{re}, \tag{13}$$

where Θ denotes all parameters need to be trained, including $\mathbf{W}_t^{(l)}$, $\mathbb{W}$, $\mathbb{W}_f$, $\mathbb{W}_i$, $\mathbb{W}_o$, $\mathbb{W}_c$, b, b_f, b_i, b_o, b_c, ϕ, and θ.

Algorithm 1 Pseudo-code of MUTANT framework under the guidance of loss function

Input: Multivariate time series $\mathbf{X}$, parameters n, τ, k.
Output: The value of loss.
1: **for** t=τ to n **do**
2: W_t = $\mathbf{X}[t-\tau+1:t]$
3: $\rho \leftarrow$ Pearson(W_t)
4: $\mathcal{G}_t \leftarrow k$-NN
5: $\tilde{\mathbf{x}}_t \leftarrow$ GCN($\mathcal{G}_t$)
6: $\tilde{\mathbf{x}}_t \leftarrow$ LSTM based Attention($\vec{\mathbf{x}}_t$)
7: $\hat{\mathbf{x}}_t \leftarrow$ VAE($\tilde{\mathbf{x}}_t$)
8: **end for**
9: Calculate $\mathcal{L}_{joint}$ using Eq. (13)
10: Back propagation and update parameters in MUTANT
11: Return $\mathcal{L}_{joint}$

Algorithm 1 shows the main process of our MUTANT. In each time window, we first construct the feature graph $\mathcal{G}_t$ (lines 3–4) and obtain the embedding vectors through the temporal GCN

Table 2
The statistics of benchmark datasets. (%) is the ratio of anomalies in each test set.

Dataset	Train	Test	#entities	#dimensions	Anomalies (%)
MSL	58,317	73,729	27	55	10.72
SMAP	135,183	427,617	55	25	13.13
SWaT	495,000	449,919	1	51	11.98
WADI	784,571	172,801	1	123	5.99

(line 5). Then we lean the weights for variables through the LSTM-based attention module, and obtain $\tilde{\mathbf{x}}_t$ (line 6). Next we use VAE to reconstruct $\tilde{\mathbf{x}}_t$, and obtain the reconstructed value $\hat{\mathbf{x}}_t$ (line 7). Finally, the model is iteratively optimized using the calculated loss function in Eq. (13)

4.4. Online detection

After training the MUTANT, we can use it to identify whether an observation at a time point t in the MTS (denoted as $\mathbf{x}_t$) is anomalous or not. Notice that the input of MUTANT is a time window W_t, instead of $\mathbf{x}_t$, and output is reconstructed input, *i.e.*, $\hat{\mathbf{x}}_t$. The difference between $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{x}}_t$, *i.e.*, reconstruction error, is adopted as the anomaly score in our model, which is denoted as $\mathbf{s}_t$, *i.e.*, $\mathbf{s}_t = (\tilde{\mathbf{x}}_t - \hat{\mathbf{x}}_t)$. A small anomaly score means the input W_t can be well reconstructed. Because we train our MUTANT on the normal data, the model learns how to reconstruct normal data and also successfully captures the hidden normal patterns in MTS. Therefore, when the reconstruction error of an observation is small, it means that it conforms to these patterns. On the contrary, a higher reconstruction error means it does not fit these patterns, and it may be an outlier. Namely, if an observation has a higher anomaly score, it is more likely to be an anomaly.

4.5. Time complexity analysis

We now analyze the time complexity of our proposed MUTANT for detecting anomalies in MTS. MUTANT is mainly composed of three sub-modules including the temporal GCN module for acquiring the relationship of variables, the LSTM-based attention module for learning the importance of variables based on time dependencies in the time dimension, and the VAE module for obtaining the normal mode of MTS. For the temporal GCN, we use GCN to aggregate first-order neighbors' features. Therefore, the time complexity of temporal GCN is $O(N_e d)$, where N_e is the number of edges in feature graph $\mathcal{G}$, d is the dimension of embedding. For LSTM-based attention, the computational complexity of LSTM is $O(nm^2)$, where n is the length of MTS, m is the number of variables. The VAE's computational complexity is $O(n^2m^2)$. Therefore, the total time complexity of MUTANT is $O(N_e d + nm^2 + n^2m^2)$.

5. Experiment

In this section, we first introduce the details of four evaluation datasets and the competitive algorithms. We study the effectiveness of our proposed MUTANT on four datasets compared to state-of-the-art baseline methods, and a hypothesis test is designed to prove the performance significance of the proposed MUTANT. We then focus on the ablation study to verify the effect of each component in our model. Finally, the parameter sensitivity and robustness of our model are discussed.

5.1. Datasets

We conduct extensive experiments on four publicly available real-world datasets.

- **Mars Science Laboratory (MSL) rover** and **Soil Moisture Active Passive (SMAP) satellite**[1] are two public real-world expert-labeled datasets from NASA [4]. Each dataset contains a training set and a test set, and anomalies in the testing set are labeled. They contain the data of 27 and 55 entities each monitored by 55 and 25 metrics (variables), respectively.
- **Secure Water Treatment (SWaT) dataset**[2] is collected from an industrial water treatment plant that produces filtered water [42]. The dataset [43] contains 11 days of continuous operations, including 7 days under normal conditions and 4 days under attack scenarios.
- **Water Distribution (WADI) dataset**[2] is collected from the WADI testbed, which consists of 16 days of continuous operation: 14 days under normal conditions and 2 days with attack scenarios.

The detailed statistics of four datasets are summarized in Table 2.

5.2. Evaluation metrics

We use *Precision*, *Recall*, and *F1-score* to evaluate the performance of our proposed model and baselines, which are defined as:

$$\begin{aligned} Precision &= \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN}, \\ F1-score &= \frac{2 \times Precision \times Recall}{Precision + Recall}, \end{aligned} \tag{14}$$

where *TP* is the True Positives, *FP* is the False Positives, and *FN* is the False negatives.

In reality, anomalies often last for a short period of time, and there will be continuous anomalies in MTS. Therefore, a point-adjusted way to measure the detection performance is proposed in [27]. This method proposes that if any observation in the ground truth anomaly segment is detected as abnormal, then the ground truth anomaly segment is considered to be successfully detected, and all observations in the segment are considered to be correctly detected outliers. In this work, the point-adjusted method is adopted to calculate the evaluation metrics.

5.3. Baselines

We compare our MUTANT against the following baselines:

- **LSTM-NDT** [4] - An LSTM-based prediction network that determines anomalies based on the prediction error. A pruning strategy is proposed to improve the accuracy of detection.
- **LSTM-VAE** [5] - A reconstruction-based model that replaces the feed-forward network in the VAE with LSTM to capture the temporal dependence of the data.
- **OmniAnomaly** [8] - A VAE-based reconstruction model that uses GRU to obtain temporal dependencies, and adopts stochastic variable connection and planar normalizing flow to improve detection accuracy.
- **USAD** [29] - An unsupervised framework with one encoder and two decoders, whose learning is inspired by GANs to increase the gap between normal data and abnormal data.

[1] https://github.com/khundman/telemanom.
[2] https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/.

Table 3
Anomaly detection accuracy in terms of precision, recall, and F1-score on four datasets.

Method	MSL			SMAP			SWaT			WADI		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
LSTM-NDT	0.9690	0.6932	0.8082	0.8455	0.9096	0.8764	0.6572	0.7754	0.7114	0.6098	0.5281	0.5660
LSTM-VAE	0.9147	0.7486	0.8234	0.7164	0.9875	0.8304	0.7123	0.9258	0.8051	0.8302	0.5203	0.6397
OmniAnomaly	0.9269	0.8502	0.8869	0.9640	0.7418	0.8384	0.9623	0.7432	0.8387	0.3017	0.9486	0.4578
USAD	0.8710	0.9536	0.9104	0.7697	0.9831	0.8634	0.9334	0.7572	0.8361	0.6451	0.3220	0.4296
MTAD-GAT	0.9374	0.8801	0.9078	0.9029	0.8997	0.9013	0.9662	0.7491	0.8439	0.3242	0.8706	0.4725
GDN	0.9050	0.8052	0.8522	0.7685	0.8591	0.8113	0.9935	0.6812	0.8082	0.9750	0.4019	0.5692
ELM-AD	0.8124	0.8603	0.8356	0.9242	0.7673	0.8384	0.8529	0.7688	0.8087	0.6021	0.5137	0.5543
InterFusion[b]	0.8853	0.9073	0.8962	0.9515	0.8481	0.8968	\	\	0.9280[b]	\	\	0.9103[b]
DAEMON[a]	0.910[a]	1.0[a]	0.953[a]	0.929[a]	0.892[a]	0.910[a]	0.966[a]	0.929[a]	0.947[a]	\	\	\
AMSL	0.9559	0.8756	0.9139	0.9431	0.9218	0.9323	0.9528	0.9452	0.9489	0.8847	0.9402	0.9116
MUTANT	0.9571	0.9709	**0.9640**	0.9788	0.9719	**0.9753**	0.9607	0.9912	**0.9757**	0.9172	0.9703	**0.9430**

[a]Results are reported in [10].
[b]Results are reported in [19].

- **MTAD-GAT** [21] - A reconstruction-based model that treats each univariate time series as a separate feature and leverages two parallel GAT layers to capture temporal and spatial dependencies, respectively.
- **GDN** [20] - An approach based on graph attention neural network which learns the relationships between different sensors, and judges abnormality according to whether it deviates from these relationships.
- **ELM-AD** [32] - A cluster-based anomaly detection method for multivariate time series. It combines the multivariate ELM-MI framework with the dynamic kernel selection method and determines the kernel in ELM-MI through clustering.
- **DAEMON** [10] - A model that combines autoencoder and adversarial training to learn the normal pattern of MTS, and thereafter uses the reconstruction error to detect anomalies.
- **InterFusion** [19] - An unsupervised method that simultaneously models the inter-metric and temporal dependency for MTS anomaly detection.
- **AMSL** [30] - A self-supervised multivariate time series anomaly detection model, which improves the generalization ability of the model through a convolutional autoencoder.

Notice that the source code of DAEMON is not publicly available, thus we directly adopt the experimental results reported in their paper [10]. Although the code of InterFusion is available, its running time is very long. For example, it used more than 100 h on the small dataset MSL. Therefore, we directly adopt the reported *F1-score* results on the large-scale datasets SWaT and WADI in their paper [19].

5.4. Experimental setting

In our experiments, we use the training set (shown in Table 2) to train detection models and use the test set (shown in Table 2) to evaluate the performance of all models.

For MUTANT, we set the number of graph neural layers l to 3, embedding dimensionality of variables to 5, τ to 20, k to 4, α to 1, and the dimension of latent representation z to 80 on SMAP, 100 on MSL, SWaT and 120 on WADI. We use the source code released by their authors for baseline evaluation. Specifically, we use the parameter settings provided in their paper, and the parameters of all baselines are tuned to be optimal. In experiments, we repeat each experiment 10 times for all methods to report average results. The source code is available at https://github.com/Coacsyf/MUTANT.

5.5. Overall performance

We compare MUTANT with eight unsupervised methods for the detection of MTS anomalies to demonstrate the overall performance of our MUTANT. Table 3 shows the obtained performance results for all methods on the four public datasets, where the best results are shown in bold. To fully demonstrate the performance of all baseline methods, for each method, we test all possible anomaly thresholds and report the highest *F1-score*.

As we can see, MUTANT significantly outperforms all baseline methods in terms of *F1-score* on all tested datasets. More specifically, experimental results indicate that MUTANT achieves average gains of 3.74% *F1-score* in comparison to the best-performed baseline across all tested datasets, reaching up to 7.18% on SMAP. MUTANT even achieves average gains of 4.09% as compared to state-of-the-art AMSL on the four datasets. Considering that the average performance gain in MTS anomaly detection reported in recent works [21,29] is usually around 1.14–1.4% in *F1-score*, this performance improvement achieved by our MUTANT is significant. This is because our MUTANT effectively captures the connections between different variables in MTS and their importance in different time windows for reconstruction-based anomaly detection through our designed modules.

LSTM-NDT, LSTM-VAE, or OmniAnomaly, which combine VAE and RNN variants, only consider the time dependence on time series and ignore the potential connections between variables in MTS. Although adversarial training is used to amplify the reconstruction error of inputs containing anomalies in auto-encoder based USAD, it ignores both the time dependencies in the time dimension and connections between variables. It can be seen that these methods present relatively good performance on datasets with only dozens of variables (*i.e.*, MSL, SMAP, and SWaT), but perform poor results on the dataset with hundreds of variables (*i.e.*, WADI).

Both MTAD-GAT and GDN are GNN-based anomaly detection methods, which consider the dynamic connections between multiple variables in MTS. In addition, MTAD-GAT also combines the forecast-based model and the reconstruction-based model to detect anomalies, while GDN leverages only the forecasting model for anomaly detection. However, they all ignore that the impact (importance) of different variables in detecting anomalies is different. Results show that MTAD-GAT achieves acceptable performance on MSL and SMAP datasets, while GDN presents a lower performance on MSL and SMAP datasets. Since ELM-AD is a cluster-based anomaly detection method, it weakens the dependence of data on time and ignores the relationship between variables, making the detection result not ideal.

Additionally, according to the experimental results reported in [10], our MUTANT is also significantly better than state-of-the-art DAEMON on MSL, SMAP, and SWaT datasets. Specifically,

Table 4
Hypothesis testing in terms of precision, recall, and F1-score on four datasets.

Method	MSL			SMAP		
	Precision	Recall	F1-score	Precision	Recall	F1-score
LSTM-NDT	1.507e−3	1.921e−17	2.028e−13	1.542e−5	5.570e−7	5.623e−11
LSTM-VAE	2.057e−8	2.351e−10	1.282e−12	4.268e−7	6.101e−5	1.113e−9
OmniAnomaly	8.237e−8	9.563e−10	2.371e−11	3.549e−5	2.900e−6	4.434e−10
USAD	4.680e−12	3.265e−11	2.064e−12	9.471e−8	1.148e−5	1.880e−11
MTAD-GAT	2.402e−7	1.681e−6	5.302e−13	7.725e−8	4.739e−6	1.389e−10
GDN	3.471e−11	2.622e−14	1.719e−15	9.464e−11	2.693e−9	5.133e−12
ELM-AD	1.843e−6	2.005e−6	3.059e−12	9.086e−8	1.139e−7	8.491e−14
InterFusion	7.298e−9	3.069e−7	5.541e−12	5.553e−3	2.260e−7	2.179e−9
AMSL	6.498e−3	4.320e−4	2.517e−8	1.051e−5	3.095e−5	3.292e−9
Method	SWaT			WADI		
	Precision	Recall	F1-score	Precision	Recall	F1-score
LSTM-NDT	3.185e−11	1.109e−9	3.724e−13	6.315e−14	1.806e−14	1.651e−17
LSTM-VAE	1.030e−5	2.086e−4	4.481e−11	1.244e−6	3.336e−13	4.743e−12
OmniAnomaly	7.708e−5	3.501e−6	2.990e−16	1.078e−7	4.454e−5	9.041e−15
USAD	2.646e−6	4.592e−8	1.988e−14	1.845e−11	2.339e−10	3.716e−14
MTAD-GAT	4.375e−4	1.286e−7	8.428e−15	2.305e−7	4.144e−6	9.026e−18
GDN	2.713e−3	2.468e−7	8.214e−16	9.529e−3	3.851e−6	2.378e−17
ELM-AD	7.494e−8	6.939e−8	1.059e−13	1.383e−7	9.298e−7	1.486e−18
InterFusion	\	\	\	\	\	\
AMSL	4.408e−6	1.119e−4	3.535e−8	1.418e−3	3.882e−3	1.193e−5

MUTANT achieves an average 5.1% improvement in terms of *F1-score* over DAEMON on large-scale MTS data, *i.e.*, SMAP and SWaT. Furthermore, MUTANT also performs better than InterFusion on all tested datasets. In particular, MUTANT achieves average gains of 4.37% as compared to InterFusion on SWaT and WADI according to the reported *F1-score* in [19].

In summary, MUTANT achieves the best detection performance on either multi-entity MTS with dozens of variables or single-entity MTS with hundreds of variables, suggesting that our proposed model effectively captures the correlations between variables in MTS and learns the reasonable importance of these variables in each time window for reconstruction-based anomaly detection.

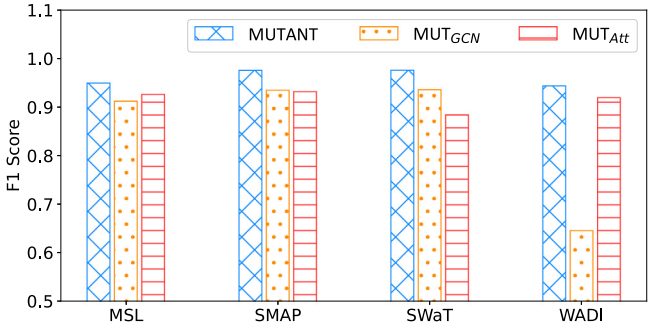


Fig. 3. The performance comparison of variants on four datasets.

5.6. Hypothesis testing

To further demonstrate the superiority of MUTANT, we conduct hypothesis testing experiments. Specifically, we statistically evaluate MUTANT against all baselines on four datasets via t-test. For each pair of comparison methods, we define null hypothesis $\mathcal{H}_0$ and the alternative hypothesis $\mathcal{H}_1$:

$$\begin{aligned} \mathcal{H}_0 &: A \approx B \\ \mathcal{H}_1 &: A < B \end{aligned} \tag{15}$$

where A is the experimental result of one of the above-mentioned baselines on a certain dataset, and B is the detection result of MUTANT in the corresponding dataset. We compute the p-value for each test and check the hypothesis at $p = 0.05$ significance level. We perform the t-test on the three evaluation metrics: *precision*, *recall* and *F1-score*, and the specific statistical evaluation results are shown in Table 4. Notice that the source code of DAEMON is not publicly available, thus we cannot do the hypothesis test with this method. Since InterFusion's running time is very long, we directly adopt the reported F1-score results on the large-scale datasets SWaT and WADI in their paper. Therefore, we also do not take the hypothesis test with InterFusion on SWaT and WADI datasets.

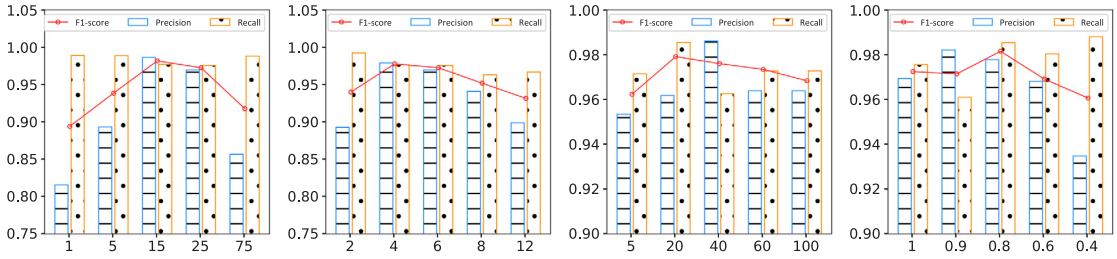
As shown in Table 4, all t-test results of *precision*, *recall* and *F1-score* are significant at $p = 0.05$. Especially in *F1-score*, MUTANT significantly outperforms all the baseline methods at $p = 0.05$. Thus we can reject the null hypothesis $\mathcal{H}_0$ and accept alternative Hypothesis $\mathcal{H}_1$. Namely, the detection performance of our proposed MUTANT is significantly better than those of all baselines. This experiment also demonstrates that the improvement of our MUTANT over state-of-the-art baselines is statistically significant in detecting anomalies for MTS.

5.7. Ablation study

To verify each component of MUTANT, we further conduct the ablation study. We compare our model with two carefully designed variations. Despite the changed part(s), all variations have the same frame structure and parameter settings. The performance of all variations in terms of *F1-score* on four datasets are shown in Fig. 3.

- **MUT**$_{GCN}$ - This variation removes the temporal GCN module, and directly uses time window W_t as the input of the reconstruction module.
- **MUT**$_{Att}$ - In this variation, we remove the LSTM-based attention mechanism, and directly feed the learned embeddings of variables into VAE for reconstruction.

Effect of temporal GCN. The comparison between MUT$_{GCN}$ and MUTANT highlights the effectiveness of the temporal GCN in MTS anomaly detection. From Fig. 3, we can observe that MUT$_{GCN}$ performs worse than MUTANT on all datasets, and even performs the worst on the WADI dataset. Specifically, MUTANT significantly improves 46.02% over MUT$_{GCN}$ in terms of *F1-score* on the dataset with hundreds of variables (*i.e.*, WADI). This indicates that our model with GCN, considering the potential connections between

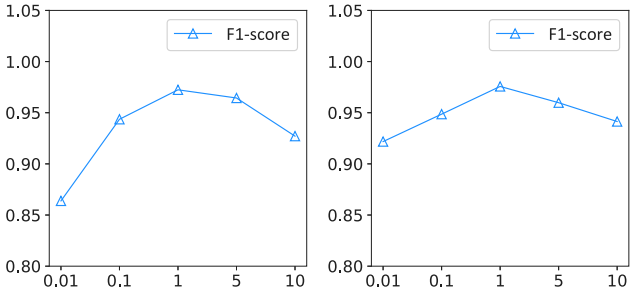


(a) Window length τ (b) k-NNs (c) Dimension of $\mathbf{z}$ (d) Size of training set

Fig. 4. Experimental results of MUTANT *w.r.t.* parameters on the SMAP dataset.

variables in MTS, works much better on the high-dimensional MTS than the low-dimensional MTS. That is, considering the connections between variables is essential for high-dimensional MTS anomaly detection.

Effect of LSTM-based attention. The comparison between MUT_{Att} and MUTANT reflects the importance of the LSTM-based attention module for MTS anomaly detection. As shown in Fig. 3, MUT_{Att} produces worse results than MUTANT on all datasets, reducing 9.4% performance in terms of *F1-score* on SWaT, which demonstrates the crucial role of our designed LSTM-based attention mechanism in learning the importance of each variable in each time window for MTS anomaly detection.



(a) SMAP (b) SWaT

Fig. 5. Experimental results of MUTANT *w.r.t.* hyper-parameter α.

5.8. Parameter sensitivity

We now investigate the sensitivity of our MUTANT *w.r.t.* the important parameters, including time window length τ, the number of nearest neighbors k, representation of latent layer $\mathbf{z}$, and size of training set. All experiments are conducted using the SMAP dataset, and the results are depicted in Fig. 4.

Fig. 4(a) shows the obtained results of MUTANT in terms of *precision*, *recall*, and *F1-score* by varying the length of time window from 1 to 75. It can be seen that the detection effect of MUTANT increases first and then decreases, as the length of the time window increases. When $\tau = 15$, MUTANT achieves the best performance. This is because when the time window is too small, due to the limited data in the window, the correlation between the variables cannot be well captured. But when the time window is too large, it will contain too much complex information, and the inter-relationship between the variables becomes complicated and thus is not properly modeled.

The second parameter we study is how MUTANT responds to different k nearest neighbors. The value of k determines how many variables similar to the current variable are selected as neighbor nodes by connecting an edge. Fig. 4(b) presents the performance of MUTANT *w.r.t.* different choices of k. We can see that the effect of k on our model has similar observations with time window length τ. The model performance achieves the best when the value of k reaches 4. Only by selecting the appropriate k can we achieve the purpose of obtaining the potential connections between different variables. If the value of k is too small, the connections between some variables may be ignored, while when the value of k is too large, more connections are established between variables. But the relationship strength between the variables will be weakened by some noise, which does not reflect the real connections of variables.

From the results in Fig. 4(c), we can see that the performance of MUTANT gradually rises and then decreases slightly as the dimension of latent layer $\mathbf{z}$ increases and achieves the best performance when dimension J reaches at 20. The main reason is that if dimension J of the latent layer $\mathbf{z}$ is too small, it is difficult to retain the essential characteristics of the time series, *i.e.*, too much information is lost, and it is not easy to be reconstructed, resulting in the overall error of normal samples is be too large. Conversely, a larger dimension causes the reconstruction model to fail to capture the essential features, making our model unable to effectively distinguish abnormal samples from normal samples.

We next study the influence of the size of the training set on the detection performance. Fig. 4(d) illustrates the performance of our MUTANT with respect to the size of the training set. Specifically, we vary the ratio γ of the original training set for each experiment, for example, if γ is set to 0.9, we remove the first 10% of the training set and use the rest as the real training set. We can observe that the performance of our MUTANT remains stable when γ varies from 1 to 0.6. Namely MUTANT is robust relative to the size of the training set. Even with 40% of the training set, our model still achieves 0.9606 in *F1-score*, which is much higher than all other comparison algorithms.

Finally, we evaluate the impact of hyper-parameter α on the detection performance of MUTANT on SMAP and SWaT datasets. α is used to balance the importance of representation learning of temporal GCN and reconstruction module. The experimental results are shown in Fig. 5. As we can see, *F1-score* of MUTANT first increases to the maximal values and then decreases as hyper-parameter α increases. This is intuitive because both temporal GCN and reconstruction module are essential for precise detection as verified in the ablation study. As shown in Fig. 5(a), *F1-score* of MUTANT reaches maximum values when α falls around 1 on SMAP. Similarly, MUTANT achieves the best performance when α is set to 1 on SWaT in Fig. 5(b). This also suggests both our proposed representation learning and reconstruction module contribute a lot to the overall performance.

5.9. Robustness evaluation

In the process of data collection, due to the influence of external factors, some wrong data may be generated or mixed with some noise data, which often reduces the accuracy of detection.

Table 5
Anomaly detection results of MUTANT on contaminated training data.

ε%	MSL			SMAP			SWaT			WADI		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
1%	0.9563	0.9655	0.9609	0.9770	0.9685	0.9727	0.9735	0.9716	0.9725	0.9416	0.9287	0.9351
5%	0.9317	0.9840	0.9570	0.9391	0.9809	0.9595	0.9749	0.9232	0.9483	0.9195	0.9480	0.9335
10%	0.9284	0.9644	0.9461	0.9386	0.9813	0.9595	0.9654	0.9298	0.9473	0.8783	0.9635	0.9189
15%	0.9097	0.9968	0.9513	0.9413	0.9886	0.9643	0.9660	0.9428	0.9543	0.9528	0.9017	0.9265
20%	0.9062	0.9968	0.9493	0.9347	0.9880	0.9606	0.9817	0.9160	0.9477	0.9496	0.8761	0.9114

Therefore, the outlier detection models must have good robustness with noise data. To evaluate the robustness of MUTANT, we design the following experiment: we use generated random numbers to replace the original data in the training set so that the training set contains some noise. More specifically, we generate ε% random numbers of the training set and replace ε% values in the training set by randomly selecting time points and variable dimensions. It is worth noting that these random numbers do not obey any distribution in order to be closer to reality.

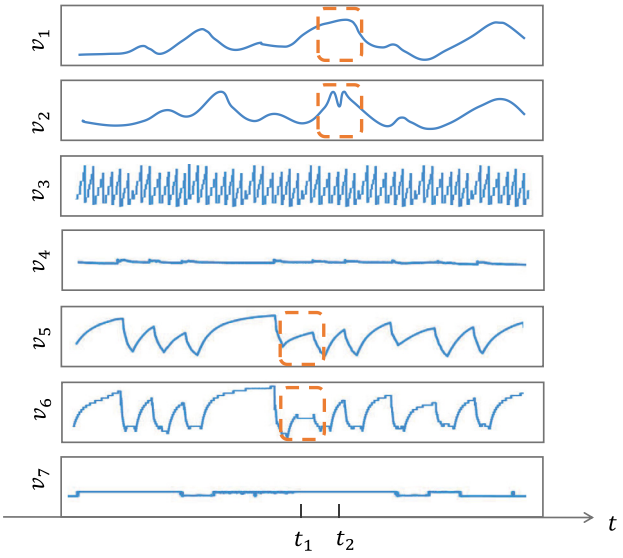
Table 5 shows the experimental results of MUTANT with different contaminated training sets on four datasets. We can observe that the performance of MUTANT decreases slightly with the increase of noise ratio ε. However, the average *F1-score* of MUTANT is only reduced by 2.31% even when noise accounts for 20%, which is still better than the performance of all baselines with original training sets shown in Table 3. The possible reason is that our MUTANT uses the variable features in each time window to construct the feature graph, and uses the embedding vectors of variables to replace the previous contaminated features to learn normal patterns in the reconstruction module, which effectively enhances the robustness of MUTANT to noise.

5.10. Case study

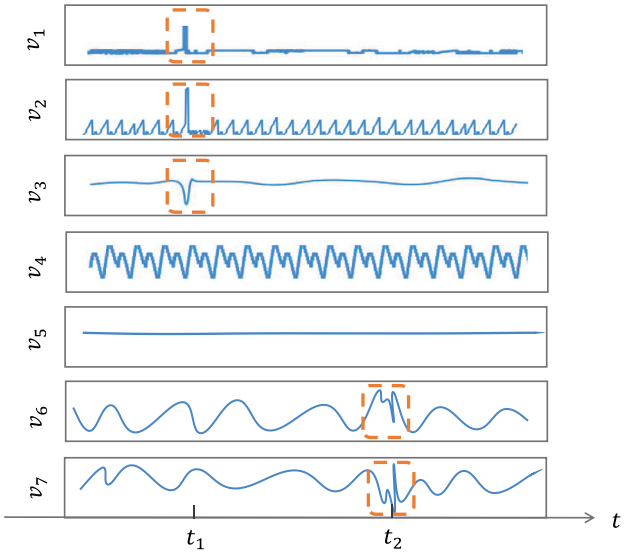
In this section, we investigate the effectiveness of the proposed model in detecting anomalies through a case study approach. In particular, we select two representative cases on MSL and SMAP datasets, respectively. Note that although two datasets include dozens of variables, we only select seven representative variables for convenience. As shown in Fig. 6, where v_i represents the time series produced by the ith variable.

The case in Fig. 6(a) comes from the MSL dataset, and the data samples at t_1 and t_2 are the anomalies detected by our model, and they are also real anomalies. It can be seen from the figure that the changing trends of variables v_1 and v_2 and the changing trends of variables v_5 and v_6 are basically consistent, which indicates that there is a strong correlation between v_1 and v_2 and between v_5 and v_6. But at time t_1, the changing trend between v_1 and v_2 changed, and the consistency is no longer maintained, which indicates that anomalies may occur in the system. The same is true for the anomaly at time t_2. However, the time series generated by each variable has not changed drastically, and even remains within the normal range, so it is difficult for other anomaly detection models that do not consider the inter-relationship between variables to find these anomalies. The proposed model, using temporal GCNs to model variables in time windows, captures the correlations between variables well and thus can effectively detect this type of anomaly.

The case shown in Fig. 6(b) comes from the SMAP dataset, where the data samples at time t_1 and t_2 are anomalies detected by our model, and they are also real anomalies. Since the time series data generated by variables v_1, v_2, and v_3 are basically stable or change periodically, when the data changes significantly at time t_1, the proposed model and other anomaly detection models can easily detect this anomaly. However, for the abnormality at time t_2, the data generated by variables v_1, v_2, and v_3 does not change. Although the time series data generated by variables v_6 and v_7 are in a state of continuous change, the data frequency at time t_2 is accelerating. This anomaly here is successfully identified by our MUTANT based on this change, while it is not detected by other anomaly detection methods. This is because we propose an attention mechanism that assigns weights to different variables based on the time dependence of the time series data. This mechanism assigns different weights to variables in different time

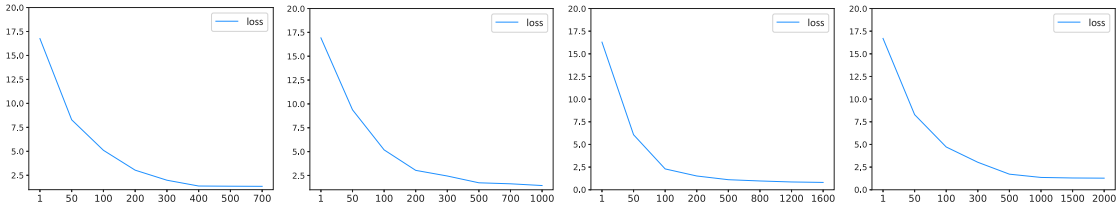


(a) A case of considering the relationship between variables.



(b) A case of considering importance of different variables.

Fig. 6. Results of case study.



(a) MSL (b) SMAP (c) SWaT (d) WADI

Fig. 7. Learning curve of loss function for MUTANT.

windows. At the time t_2, the frequency of data changes generated by v_6 and v_7 is accelerated, and our model pays more attention to these variables and gives them greater weight so that abnormality at time t_2 is detected.

5.11. Convergence of mutant

Fig. 7 shows the learning curves of the loss function of our MUTANT model on the four datasets, where the abscissa indicates the number of epochs, and the ordinate indicates the loss value. We can see that our model can converge quickly and remain stable, which reflects the high efficiency of our proposed model in this work.

6. Conclusion

In this paper, we propose a novel unsupervised method MUTANT for MTS anomaly detection. MUTANT first constructs a feature graph based on variable features for each time window in MTS and uses GCN to learn the embeddings for all variables. Then, MUTANT feeds the embeddings of variables into the proposed attention-based reconstruction module, which consists of an LSTM-based attention module that learns the importance of variables in each time window and a VAE module that learns the latent representation for each observation to capture normal patterns of MTS. Additionally, we use end-to-end training to optimize our model by a joint learning objective function. Experimental results on four public benchmark datasets demonstrate the superiority of the proposed MUTANT in comparison with state-of-the-art techniques. For future work, we are very interested in inducing a more robust self-supervised learning framework based on contrastive learning for MTS anomaly detection.

CRediT authorship contribution statement

Yunfei Shi: Methodology, Software, Data curation, Validation, Visualization, Writing – original draft. **Bin Wang:** Validation, Writing – original draft. **Yanwei Yu:** Conceptualization, Supervision, Writing – review & editing, Funding acquisition. **Xianfeng Tang:** Formal analysis, Writing – review & editing. **Chao Huang:** Writing – review & editing. **Junyu Dong:** Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets used in this work are publicly available.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China under grant Nos. 62176243, 41927805, and 61773331, and the National Key Research and Development Program of China under grant Nos. 2018AAA0100602 and 2019YFC1509100.

References

[1] R. Chalapathy, S. Chawla, Deep learning for anomaly detection: A survey, 2019, arXiv preprint arXiv:1901.03407.
[2] D.M. Hawkins, Identification of Outliers, Vol. 11, Springer, 1980.
[3] N. Laptev, S. Amizadeh, I. Flint, Generic and scalable framework for automated time-series anomaly detection, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 1939–1947.
[4] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 387–395.
[5] D. Park, Y. Hoshi, C.C. Kemp, A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, IEEE Robot. Autom. Lett. 3 (3) (2018) 1544–1551.
[6] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, S.-K. Ng, MAD-gan: Multivariate anomaly detection for time series data with generative adversarial networks, in: International Conference on Artificial Neural Networks, Springer, 2019, pp. 703–716.
[7] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, G. Shroff, LSTM-based encoder-decoder for multi-sensor anomaly detection, 2016, arXiv preprint arXiv:1607.00148.
[8] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, D. Pei, Robust anomaly detection for multivariate time series through stochastic recurrent neural network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2828–2837.
[9] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, N.V. Chawla, A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 1409–1416.
[10] X. Chen, L. Deng, F. Huang, C. Zhang, Z. Zhang, Y. Zhao, K. Zheng, DAEMON: Unsupervised anomaly detection and interpretation for multivariate time series, in: 2021 IEEE 37th International Conference on Data Engineering, ICDE, IEEE, 2021, pp. 2225–2230.
[11] Y. Yan, L. Cao, E.A. Rundensteiner, Scalable top-n local outlier detection, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 1235–1244.
[12] F. Liu, Y. Yu, P. Song, Y. Fan, X. Tong, Scalable KDE-based top-n local outlier detection over large-scale data streams, Knowl.-Based Syst. 204 (2020) 106186.
[13] E.M. Knorr, R.T. Ng, V. Tucakov, Distance-based outliers: algorithms and applications, VLDB J. 8 (3) (2000) 237–253.
[14] M.M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 2000, pp. 93–104.
[15] B. Zong, Q. Song, M.R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection, in: International Conference on Learning Representations, 2018.
[16] P. Lv, Y. Yu, Y. Fan, X. Tang, X. Tong, Layer-constrained variational autoencoding kernel density estimation model for anomaly detection, Knowl.-Based Syst. 196 (2020) 105753.
[17] R. Laxhammar, G. Falkman, E. Sviestins, Anomaly detection in sea traffic-a comparison of the gaussian mixture model and the kernel density estimator, in: 2009 12th International Conference on Information Fusion, IEEE, 2009, pp. 756–763.

[18] E. Schubert, A. Zimek, H.-P. Kriegel, Generalized outlier detection with flexible kernel density estimates, in: Proceedings of the 2014 SIAM International Conference on Data Mining, SIAM, 2014, pp. 542–550.
[19] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, D. Pei, Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 3220–3230.
[20] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 4027–4035.
[21] H. Zhao, Y. Wang, J. Duan, C. Huang, D. Cao, Y. Tong, B. Xu, J. Bai, J. Tong, Q. Zhang, Multivariate time-series anomaly detection via graph attention network, in: 2020 IEEE International Conference on Data Mining, ICDM, IEEE, 2020, pp. 841–850.
[22] M. Gupta, J. Gao, C.C. Aggarwal, J. Han, Outlier detection for temporal data: A survey, IEEE Trans. Knowl. Data Eng. 26 (9) (2013) 2250–2267.
[23] A. Kejariwal, Introducing practical and robust anomaly detection in a time series, Twitter Eng. Blog. Web. 15 (2015).
[24] D.T. Shipmon, J.M. Gurevitch, P.M. Piselli, S.T. Edwards, Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data, 2017, arXiv preprint arXiv:1708.03665.
[25] A. Siffer, P.-A. Fouque, A. Termier, C. Largouet, Anomaly detection in streams with extreme value theory, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 1067–1075.
[26] Y. Yu, P. Lv, X. Tong, J. Dong, Anomaly detection in high-dimensional data based on autoregressive flow, in: International Conference on Database Systems for Advanced Applications, Springer, 2020, pp. 125–140.
[27] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, et al., Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 187–196.
[28] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, Q. Zhang, Time-series anomaly detection service at microsoft, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 3009–3017.
[29] J. Audibert, P. Michiardi, F. Guyard, S. Marti, M.A. Zuluaga, Usad: Unsupervised anomaly detection on multivariate time series, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 3395–3404.
[30] Y. Zhang, J. Wang, Y. Chen, H. Yu, T. Qin, Adaptive memory networks with self-supervised learning for unsupervised anomaly detection, IEEE Trans. Knowl. Data Eng. (2022).
[31] Y. Jiao, K. Yang, D. Song, D. Tao, Timeautoad: Autonomous anomaly detection with self-supervised contrastive loss for multivariate time series, IEEE Trans. Netw. Sci. Eng. 9 (3) (2022) 1604–1619.
[32] X. Peng, H. Li, F. Yuan, S.G. Razul, Z. Chen, Z. Lin, An extreme learning machine for unsupervised online anomaly detection in multivariate time series, Neurocomputing 501 (2022) 596–608.
[33] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.
[35] L. Xie, D. Pi, X. Zhang, J. Chen, Y. Luo, W. Yu, Graph neural network approach for anomaly detection, Measurement 180 (2021) 109546.
[36] W.L. Hamilton, R. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.
[37] J. Lu, J. Xuan, G. Zhang, X. Luo, Structural property-aware multilayer network embedding for latent factor analysis, Pattern Recognition 76 (2018) 228–241.
[38] W. Duan, J. Xuan, M. Qiao, J. Lu, Learning from the dark: boosting graph convolutional neural networks with diverse negative samples, in: Proceedings of the AAAI Conference on Artificial Intelligence, 36, (6) 2022, pp. 6550–6558.
[39] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI Open 1 (2020) 57–81.
[40] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in: Y. Bengio, Y. LeCun (Eds.), 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.
[41] D.P. Kingma, M. Welling, Stochastic gradient VB and the variational auto-encoder, in: Second International Conference on Learning Representations, ICLR, Vol. 19, 2014, p. 121.
[42] J. Goh, S. Adepu, K.N. Junejo, A. Mathur, A dataset to support research in the design of secure water treatment systems, in: International Conference on Critical Information Infrastructures Security, Springer, 2016, pp. 88–99.
[43] A.P. Mathur, N.O. Tippenhauer, Swat: A water treatment testbed for research and training on ics security, in: 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater), IEEE, 2016, pp. 31–36.